SOLVING REDUCED KKT SYSTEMS IN BARRIER METHODS FOR LINEAR AND QUADRATIC PROGRAMMING

Philip E. GILL[*], Walter MURRAY[†],
Dulce B. PONCELEÓN[‡] and Michael A. SAUNDERS[†]

Technical Report SOL 91-7[§]
July 1991

Abstract

In barrier methods for constrained optimization, the main work lies in solving large linear systems $Kp = r$, where K is symmetric and indefinite.

For linear programs, these KKT systems are usually reduced to smaller positive-definite systems $AH^{-1}A^Tq = s$, where H is a large principal submatrix of K. These systems can be solved more efficiently, but $AH^{-1}A^T$ is typically more ill-conditioned than K.

In order to improve the numerical properties of barrier implementations, we discuss the use of "reduced KKT systems", whose dimension and condition lie somewhere in between those of K and $AH^{-1}A^T$. The approach applies to linear programs and to positive semidefinite quadratic programs whose Hessian H is at least partially diagonal.

We have implemented reduced KKT systems in a primal-dual algorithm for linear programming, based on the sparse indefinite solver MA27 from the Harwell Subroutine Library. Some features of the algorithm are presented, along with results on the *netlib* LP test set.

Keywords: linear programming, quadratic programming, indefinite systems, KKT systems, barrier methods, interior-point methods.

1. Introduction

We discuss barrier methods for solving linear and quadratic programs expressed in the standard form

$$\begin{array}{ll} \underset{x}{\text{minimize}} & c^Tx + \tfrac{1}{2}x^TQx \\ \text{subject to} & Ax = b, \quad l \le x \le u, \end{array} \tag{1.1}$$

where A is $m \times n$ ($m \le n$) and Q is symmetric positive semidefinite. The problem is a linear program (LP) when $Q = 0$, and a quadratic program (QP) otherwise.

[*]Department of Mathematics, University of California at San Diego, La Jolla, CA 92093, USA.

[†]Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, CA 94305-4022, USA.

[‡]Apple Computer, Inc., 20525 Mariani Ave, Cupertino, CA 95014, USA.

[§]The material contained in this report is based upon research supported by the National Science Foundation Grant DDM-8715153 and the Office of Naval Research Grant N00014-90-J-1242.

We assume that an optimal solution (x^*, π^*) exists, where π^* is a set of Lagrange multipliers for the constraints $Ax = b$.

Implicit within most of the current barrier or interior-point algorithms is a so-called *KKT system* of the form

$$K \begin{pmatrix} \Delta x \\ -\Delta \pi \end{pmatrix} = \begin{pmatrix} w \\ r \end{pmatrix}, \qquad K \equiv \begin{pmatrix} H & A^T \\ A & \end{pmatrix}, \tag{1.2}$$

in which $H = Q + D$ where D is positive semidefinite and diagonal. The search direction $(\Delta x, \Delta \pi)$ is used to update the current solution estimate (x, π). In some cases it is obtained by solving the same KKT system with more than one right-hand side. It is critical that such systems be solved quickly and reliably.

1.1. Reduced KKT Systems

If H is nonsingular and diagonal (as in the LP case), it is common to use it as a block pivot and reduce (1.2) to a system involving $AH^{-1}A^T$. In general this is an *unstable process* because H usually contains some very small diagonals. The main advantages are that $AH^{-1}A^T$ is much smaller than K and it is positive-definite.

Our aim is to discuss an intermediate strategy in which *part of* H is used as a block pivot. The reduced systems obtained are considerably smaller than K, and typically no more than twice as large as $AH^{-1}A^T$. The approach retains the numerical reliability of factoring the full matrix K, with an efficiency that is closer to that of using $AH^{-1}A^T$. It also provides a convenient way of dealing with *free variables* and *dense columns* (i.e., variables with bounds $-\infty \le x_j \le \infty$ and columns of A that have many nonzeros).

In the QP case when Q is at least partly diagonal, reduced KKT systems can still be formed efficiently.

The proposed use of reduced KKT systems is motivated by the sensitivity analysis in [Pon90] and by the investigation of preconditioners for KKT systems in [GMPS90]. Note from both references that *large* diagonals in H give K a deceptively high condition number, but they do not cause sensitivity in the solution of systems involving K. (Similarly, a system $Dx = b$ with $D = \mathrm{diag}(10^{20}, 10^{10}, 1, 1, 1)$ has a well-defined solution, even though $\mathrm{cond}(D) = 10^{20}$.)

In fact, large diagonals of H are *desirable*, since they are obvious candidates for a block pivot. For example, if $\|A\| \approx 1$, any diagonals H_{jj} significantly larger than one could be included in the block pivot. The associated *reduced* system reflects the true sensitivity of linear systems involving K.

2. A Primal-Dual QP Algorithm

For concreteness we describe the main parts of a primal-dual barrier algorithm for LP and QP.[1] After deriving the KKT systems to be solved, we are able to discuss certain numerical issues.

[1]Related references for LP are [Meg86, KMY88, MMS89, LMS89, LMS90, Meh89, Meh90, Meh91]. Some dealing explicitly with QP are [MAR88, MA89, AHRT90, CLMS90, JSS90, Pon90, VC91].

In order to handle upper and lower bounds symmetrically, we slightly generalize the approach of Lustig et al.[LMS89, LMS90] and restate problem (1.1) as follows:

$$\begin{array}{ll} \underset{x,\, s_1,\, s_2,\, p}{\text{minimize}} & c^Tx + \frac{1}{2}x^TQx + \frac{1}{2}\|\gamma x\|^2 + \frac{1}{2}\|p\|^2 \\ \text{subject to} & \begin{array}{rcl} Ax \qquad\qquad + \delta p & = & b, \\ x - s_1 \qquad\qquad & = & l, \\ x \qquad + s_2 \qquad & = & u, \end{array} \end{array} \tag{2.1}$$

with $s_1,\ s_2 \geq 0$. The scalars γ and δ are intended to be "small". The dual variables associated with the three sets of equality constraints will be denoted by π, z and $-y$. At a solution, z and y are non-negative.

We assume that $l < u$, since fixed variables ($l_j = u_j$) can be absorbed into b. If $l_j = -\infty$ or $u_j = +\infty$, we omit the corresponding equation in $x - s_1 = l$ or $x + s_2 = u$. (In particular, both equations are omitted for free variables.) Symmetric treatment of the bounds via s_1 and s_2 allows a problem to be treated "as it stands", without converting the bounds to $0 \leq x \leq u$ (say). The latter practice is hazardous in the case of "almost free variables", whose bounds are not large enough to be treated as infinite (e.g., $-10^6 \leq x_j \leq 10^6$).

2.1. The Barrier Subproblem

For some scalar $\mu > 0$, the associated barrier subproblem is to minimize

$$c^Tx + \tfrac{1}{2}x^TQx + \tfrac{1}{2}\|\gamma x\|^2 + \tfrac{1}{2}\|p\|^2 - \mu \sum_j \ln(s_1)_j - \mu \sum_j \ln(s_2)_j$$

subject to the same equality constraints. Optimality conditions for this subproblem include the equation $p = \delta\pi$. We can therefore eliminate p immediately. The remaining optimality conditions may be stated as the following equations:

$$f_\mu(x, s_1, s_2, \pi, z, y) \ = \ \begin{pmatrix} b - Ax - \delta^2\pi \\ l - x + s_1 \\ u - x - s_2 \\ c + Qx + \gamma^2 x - A^T\pi - z + y \\ \mu e - S_1 Z e \\ \mu e - S_2 Y e \end{pmatrix} \ \equiv \ \begin{pmatrix} r \\ t_1 \\ t_2 \\ t \\ v_1 \\ v_2 \end{pmatrix} \ = \ 0, \tag{2.2}$$

where e is a column of ones and S_1, S_2, Z, Y are diagonal matrices composed from s_1, s_2, z, y. Primal-dual methods maintain positive approximations to all of the latter vectors.

2.2. The Newton Direction

The Newton equations for generating a search direction $(\Delta x, \Delta s_1, \Delta s_2, \Delta\pi, \Delta z, \Delta y)$ are

$$\begin{array}{rcccccccl} A\Delta x & & & + & \delta^2\Delta\pi & & & = & r \\ \Delta x & -\Delta s_1 & & & & & & = & t_1 \\ \Delta x & & +\Delta s_2 & & & & & = & t_2 \\ -(Q+\gamma^2 I)\Delta x & & & + & A^T\Delta\pi & + \ \Delta z & - \ \Delta y & = & t \\ & Z\Delta s_1 & & & & +S_1\Delta z & & = & v_1 \\ & & Y\Delta s_2 & & & & +S_2\Delta y & = & v_2, \end{array} \tag{2.3}$$

which may be solved by defining

$$\begin{aligned} H_0 &= Q + S_1^{-1}Z + S_2^{-1}Y, \\ w &= S_1^{-1}(Zt_1 + v_1) + S_2^{-1}(Yt_2 - v_2) - t, \end{aligned} \tag{2.4}$$

solving the KKT-like system

$$K\begin{pmatrix} \Delta x \\ -\Delta\pi \end{pmatrix} = \begin{pmatrix} w \\ r \end{pmatrix}, \qquad K \equiv \begin{pmatrix} H_0 + \gamma^2 I & A^T \\ A & -\delta^2 I \end{pmatrix}, \tag{2.5}$$

and finally solving the equations

$$\begin{aligned} \Delta s_1 &= \Delta x - t_1, \\ \Delta s_2 &= t_2 - \Delta x, \\ S_1\Delta z &= v_1 - Z\Delta s_1, \\ S_2\Delta y &= v_2 - Y\Delta s_2. \end{aligned} \tag{2.6}$$

It is straightforward to show that any values of x and π give the same search direction $(\Delta s_1, \Delta s_2, \Delta z, \Delta y)$.

2.3. Regularization

The above definition of K shows its dependence on γ and δ. Later we shall not need H_0 itself, but will work with $H \equiv H_0 + \gamma^2 I$.

The perturbations involving γ and δ are included to "regularize" the problem. The term $\frac{1}{2}\|\gamma x\|^2$ ensures that $\|x^*\|$ is bounded, and the term δp allows $Ax = b$ to be satisfied in some least-squares sense if the original constraints have no feasible solution. An important property is that both perturbations help preserve the nonsingularity of K. For example, if A does not have full row rank, K is singular unless we choose $\delta > 0$. Similarly, suppose some columns of A associated with free variables are linearly dependent; if the corresponding columns of H_0 are also dependent (e.g. if the problem is an LP), K is singular unless we choose $\gamma > 0$. (An alternative means for preserving nonsingularity with free variables is given in [GMPS91].)

Since $p = \delta\pi$ at a solution, the regularization terms in the objective function are effectively $\frac{1}{2}\|\gamma x\|^2 + \frac{1}{2}\|\delta\pi\|^2$, which has a satisfying symmetry and shows that both x^* and π^* are bounded if γ and δ are nonzero.

Objective perturbations of the form $\frac{1}{2}\|\gamma x\|^2$ have been studied by Mangasarian et al.[MM79, Man84], who show that the LP solution is not perturbed if γ is sufficiently small. The approach has been successfully pursued in the interior-point context by Setiono [Set90b].

An alternative form of regularization is the proximal-point method of Rockafellar [Roc76], which involves an objective term of the form $\frac{1}{2}\|\gamma(x - x_k)\|^2$ and does not perturb the problem as $x_k \to x^*$ even if γ is not particularly small. Again, this approach has been successfully explored by Setiono [Set89, Set90a, Set90c].

2.4. A Predictor-Corrector Approach

A number of authors (e.g. [MAR88, KLSW89, JSS90]) have suggested alternatives to the Newton direction. Most of these suggestions may be traced to the idea of "extrapolation" first described by Fiacco and McCormick [FM68]. The algorithm we have implemented is similar to that suggested by Mehotra [Meh89, Meh90]. Implementations based on his suggestion have been shown to be efficient in practice (see [Meh89, Meh90, LMS90]).

The approach requires two solves of the Newton system (2.3) with different vectors v_1 and v_2 in the right-hand side. A *predictor step* $(\Delta\widehat{x}, \Delta\hat{s}_1, \Delta\hat{s}_2, \Delta\hat{\pi}, \Delta\widehat{z}, \Delta\widehat{y})$ is obtained by solving with $v_1 = -S_1Ze$ and $v_2 = -S_2Ye$ (i.e. $\mu = 0$ in (2.2)). A *corrector direction* is then computed using $v_1 = \mu e - S_1Ze - \Delta\widehat{S}_1\Delta\widehat{Z}e$ and $v_2 = \mu e - S_2Ye - \Delta\widehat{S}_2\Delta\widehat{Y}e$.

If the predictor step is "large" it seems possible that a poor corrector direction will result. Therefore as a precaution we sometimes scale the predictor step down before constructing the second v_1 and v_2. Let $\hat{\alpha}_x$ and $\hat{\alpha}_z$ be maximum steps along the predictor direction that keep (s_1, s_2) and (z, y) non-negative, and let $\phi = 0.1$.

If $\hat{\alpha}_x < \phi$, we consider that the predictor steps $\Delta\hat{s}_1$ and $\Delta\hat{s}_2$ are excessively large and scale them down by the factor $\tau(2-\tau)$, where $\tau = \hat{\alpha}_x/\phi$.

Similarly if $\hat{\alpha}_z < \phi$, we scale $\Delta\widehat{z}$ and $\Delta\widehat{y}$ down by the factor $\tau(2-\tau)$, where $\tau = \hat{\alpha}_z/\phi$. (Note that $0 < \tau(2-\tau) < 1$ in both cases.)

We have experimented with a few other values of ϕ but not observed any profound effect. The value $\phi = 0$ corresponds to accepting the predictor step as it stands, and $\phi = 1$ would "interfere" rather too often. The chosen value 0.1 gave slightly better performance than 0, as measured by the number of times that the corrector direction was accepted.

It can be demonstrated that the corrector direction is not necessarily a descent direction for $\|f_\mu\|^2$. We are therefore prepared to fall back on the Newton direction as described below. Different but analogous precautions are taken by Mehrotra [Meh90].

2.5. Steplengths

Given a positive vector z and a search direction Δz, we need a trial steplength α_z that keeps $z + \alpha_z\Delta z$ positive. It is common practice to define such a steplength as

$$\alpha_z = \sigma \min_{\Delta z_j < 0} z_j/|\Delta z_j|,$$

where $\sigma \in (0,1)$. Such a steplength might be written as the function $\alpha_z = \alpha(z)$, with Δz and σ known from the context. In our case we need two such steplengths,

$$\alpha_x = \alpha(s_1, s_2) \quad \text{and} \quad \alpha_z = \alpha(z, y),$$

where we mean that α_x keeps both s_1 and s_2 positive, and α_z keeps both z and y positive. Following [LMS90] we used $\sigma = 0.99995$ for the numerical results of Section 7.

Given a descent direction, the usual procedure in a descent method is to take a step along the direction that reduces some merit function. In our case it is not clear that a suitable reduction in $\|f_\mu\|^2$ can be obtained using the same step $\alpha(s_1, s_2, z, y)$ in all the variables. It has been observed in practice [LMS89, LMS90] that taking different steps in the primal and dual variables leads to fewer iterations when solving linear programs. One reason for this is clear. Suppose we are solving an LP with $\delta = \gamma = 0$. After a step α_x in the primal variables and α_z in the dual variables we have

$$r \leftarrow (1 - \alpha_x) r \quad \text{and} \quad t \leftarrow (1 - \alpha_z) t.$$

(We assume $r \neq 0$, $t \neq 0$, $\alpha_x < 1$ and $\alpha_z < 1$.) If we take the same step in the primal and dual variables and wish to remain feasible to the same extent, the required step is

$$\alpha(s_1, s_2, z, y) = \min(\alpha_x, \alpha_z),$$

and the total reduction in $\|r\|^2 + \|t\|^2$ will not be as great. After 20 or 30 iterations the reduction attained by the two strategies may be significantly different.

2.6. The Linesearch

Gill et al.[GMPS91, Section 7.1] analyze a primal-dual barrier algorithm for linear programming based on the Newton direction and separate steps in the primal and dual variables. They show that the component of the Newton direction in the primal variables is a descent direction for a merit function based on the primal barrier function.[2] It is proved prove that the iterates converge to a solution provided the step taken in the primal variables reduces this merit function. (Such a step always exists.) Moreover, from the nature of the merit function and the fact that the Newton direction is a direction of *sufficient descent*, it is clear that almost any nonzero step in the primal variables will suffice. The step in the dual variables is almost arbitrary. The only requirement is that it be bounded and that z and y be kept nonzero.

The merit function in our implementation is different from that advocated in [GMPS91] but is similar in spirit. We require that $\|f_\mu\|^2$ be reduced. We anticipate that a reduction in this function almost always implies a reduction in $M(x, s_1, s_2, \rho)$.

Briefly, if a trial step along the corrector direction reduces $\|f_\mu\|^2$, the step is taken and the iteration is complete. Otherwise, the Newton direction is computed

[2]For LP problems of the form (2.1) without regularization, the merit function would be $M(x, s_1, s_2, \rho) = c^T x - \mu \sum_j \ln(s_1)_j - \mu \sum_j \ln(s_2)_j + \rho(\|b - Ax\|_1 + \|l - x + s_1\|_1 + \|u - x - s_2\|_1)$.

and used to reduce $\|f_\mu\|^2$ (perhaps with the aid of a back-tracking linesearch). Convergence is assured if K remains sufficiently nonsingular.

More precisely:

1. Separate steplengths $\alpha_x = \alpha(s_1, s_2)$ and $\alpha_z = \alpha(z, y)$ are computed for the corrector direction. If these reduce the merit function $\|f_\mu\|^2$ by a sufficient amount, they are accepted and the iteration is complete.

2. Otherwise, the Newton direction (2.3) is computed along with new steplengths α_x and α_z. If the merit function is sufficiently reduced, the steps are taken and the iteration is complete.

3. Otherwise, the steplengths are made equal (to the smaller of the two) and the merit function is tested again. If necessary, the steplengths are repeatedly halved until the merit function is suitably reduced.

4. If up to 5 halvings of the steplengths fail to reduce the merit function, we assume that the search direction has insufficient accuracy. The linesearch terminates with a request for stricter tolerances in factorizing the KKT system (see Section 4). In practice, this is perhaps the most important function of the linesearch.

2.7. Reducing the Barrier Parameter μ

The initial and minimum values of μ are to some extent user-specifed; see Section 6. At intervening iterations, μ is reduced to $(1-\alpha_\mu)\mu$, where $\alpha_\mu = \min(\alpha_x, \alpha_z, \sigma)$. Thus, μ decreases monotonically and more rapidly if larger steps are taken.

This choice of μ is simple and does not depend on the duality gap (which is difficult to define for infeasible points). It appears to be satisfactory in practice.

3. Reduced KKT systems

If $H = H_0 + \gamma^2 I$ is nonsingular, (2.5) may be solved using the *range-space equations* of optimization:

$$(AH^{-1}A^T + \delta^2 I)\Delta\pi = r - AH^{-1}w, \qquad H\Delta x = A^T\Delta\pi + w. \tag{3.1}$$

The main benefit is that $AH^{-1}A^T + \delta^2 I$ is much smaller than K and is positive definite, so that well-established sparse Cholesky factorizations can be applied. Some drawbacks are as follows:

1. $AH^{-1}A^T + \delta^2 I$ is normally more ill-conditioned than K.

2. Free variables complicate direct use of (3.1) by making H_0 singular.

3. Relatively dense columns in A degrade the sparsity of $AH^{-1}A^T + \delta^2 I$ and its factors.

In general, small diagonals of H prevent it from being a "good" block pivot from a numerical point of view. That is, if Gaussian elimination were applied to K, not all diagonals of H would be acceptable as pivots. To overcome this difficulty, we note that in practice *most of* H is likely to be acceptable as a block pivot. Thus we partition H and A as

$$H = \begin{pmatrix} H_N & \\ & H_B \end{pmatrix}, \qquad A = \begin{pmatrix} N & B \end{pmatrix},$$

where the diagonals of H_N are a "reasonable" size compared to the nonzeros in the corresponding columns of N. In general, a reordering of the variables is implied. The column dimension of N may be anywhere from 0 to n (and similarly for B). The KKT system (2.5) becomes

$$\begin{pmatrix} H_N & & N^T \\ & H_B & B^T \\ N & B & -\delta^2 I \end{pmatrix} \begin{pmatrix} \Delta x_N \\ \Delta x_B \\ -\Delta\pi \end{pmatrix} = \begin{pmatrix} w_N \\ w_B \\ r \end{pmatrix}, \tag{3.2}$$

which may be solved via the *reduced KKT system*

$$K_B \begin{pmatrix} \Delta x_B \\ -\Delta\pi \end{pmatrix} = \begin{pmatrix} w_B \\ r - N H_N^{-1} w_N \end{pmatrix}, \qquad K_B \equiv \begin{pmatrix} H_B & B^T \\ B & -N H_N^{-1} N^T - \delta^2 I \end{pmatrix}. \tag{3.3}$$

The final step is to solve the typically diagonal system $H_N \Delta x_N = w_N + N^T \Delta\pi$.

If H_N happens to be all of H, $K_B = -AH^{-1}A^T - \delta^2 I$ and the reduced KKT system is equivalent to (3.1). Otherwise, K_B is a symmetric indefinite matrix. Like K it can be processed by a sparse indefinite solver such as MA27 [DR82, DR83] (an implementation of the factorization described in [BP71, BK77]). The above difficulties are resolved as follows:

1. K_B need not be more ill-conditioned than K.
2. Free variables are always included in B.
3. Dense columns of A are also included in B. (They do not necessarily degrade the sparsity of the K_B factors.)

As with current Cholesky solvers, MA27 has an *Analyze* phase (to choose a row and column ordering for K_B and to set up a data structure for its factors) and a *Factor* phase (to compute the factors themselves). Some disadvantages of working with reduced KKT systems are:

1. Whenever the N–B partition is altered, a new *Analyze* is required. This is usually less expensive than the *Factor* phase, often by an order of magnitude, and we expect it to be needed only sometimes (typically the last few iterations). However, it can be costly for certain structures in A.
2. To date, sparse indefinite factorization is more expensive than Cholesky factorization when there is a large degree of indefiniteness (as in KKT systems).

3.1. Related Work

To keep H nonsingular in the presence of free variables, some authors have treated each such variable as the difference between two nonnegative variables. Lustig, et al.[LMS89, LMS90] report satisfactory performance on the *netlib* LP test set, which contains a few relevant examples. To deal with problems involving *many* free variables, others authors have introduced moving artificial bounds (e.g. [Marx89] and Vanderbei [Van90a], who cites difficulties with the first approach).

Dense columns have been handled by using Cholesky factors of the sparse part of $AH^{-1}A^T$ to precondition the conjugate-gradient method (e.g. [GMSTW86]) or to form a certain Schur complement [Marx89, LMS89]. A difficulty is that the "sparse" Cholesky factor usually becomes even more ill-conditioned than the one associated with all of $AH^{-1}A^T$. More recently, the approach of *splitting* or *stretching* dense columns has been proposed and implemented with success [LMC89, Van90b]. The increased problem size is perhaps an inconvenience if not a difficulty. (See also Grcar [Grc90], who recommends the term *stretching* and gives an extremely thorough development and analysis of this new approach to solving sparse linear equations.)

Further recent work on solving large indefinite systems (to avoid the difficulties associated with $AH^{-1}A^T$) appears in [Tur90, Meh91, FM91, VC91]. Iterative solution of the full KKT system is explored in [GMPS90].

3.2. Quadratic Programs

In general, the Hessian of a convex quadratic objective will have the form

$$Q = \begin{pmatrix} 0 & & \\ & \bar{D} & E^T \\ & E & \bar{Q} \end{pmatrix}, \tag{3.4}$$

where $\bar{D}$ is diagonal and positive definite, and the full Q is positive semidefinite. The dimensions of each partition depend on the application and could be anything from 0 to n.

The associated matrix $H = Q + S_1^{-1}Z + S_2^{-1}Y$ has the structure

$$H = \begin{pmatrix} \hat{H} & & \\ & \widehat{D} & E^T \\ & E & \widehat{Q} \end{pmatrix}, \tag{3.5}$$

where $\hat{H}$ and $\widehat{D}$ are diagonal. Reduced KKT systems can be formed as before, using suitably large diagonals in $\operatorname{diag}(\hat{H}, \widehat{D})$ as a block pivot. Variables associated with $\widehat{Q}$ are most easily dealt with by keeping them in the "B" partition of K (alongside those associated with dense columns of A).

A diagonal of $\operatorname{diag}(\hat{H}, \widehat{D})$ may be judged "suitably large" by comparison with the corresponding column of A. (Since Q and H are semidefinite, there is no need to compare a diagonal of $\widehat{D}$ with the corresponding columns of E.)

3.3. Separable QP

To match the OB1 implementation based on $AH^{-1}A^T$ [LMS90], Carpenter et al.[CLMS90] have emphasized the case where $\bar{Q}$ and E are null and Q is purely diagonal. Any convex QP can be transformed into this case by using the Cholesky factorization $P\bar{Q}P^T = LL^T$, where P is a permutation matrix and L is lower triangular (and possibly trapezoidal). Introducing linear constraints of the form

$$x_L = L^T P x_Q$$

leads to a larger problem of the required form.

Such an approach may often be satisfactory, especially if the dimension of $\bar{Q}$ is relatively low. However, some implementations based on $AH^{-1}A^T$ would have difficulty with the additional free variables x_L. Even if these are dealt with "correctly" via the KKT system, the factors of the reduced KKT systems (which now involve L) are likely to be more dense than in the preceding direct approach.

Further discussion of this subject, along with numerical comparisons, appears in [VC91].

In summary, the use of reduced KKT systems for structured sparse QP Hessians (3.4) provides greater flexibility, and hence greater efficiency in at least some cases, than use of the full KKT system [Pon90] or the fully reduced system $AH^{-1}A^T$ [CLMS90].

4. Stability and Sparsity Tolerances

Let $S = -(NH_N^{-1}N^T + \delta^2 I)$ be the Schur complement appearing in the south-east corner of K_B (3.3). (We assume H_N is diagonal.) In our implementation, three parameters are used to control the choice of N and the factorization of K_B, with typical values as follows:

$$\begin{aligned} \mathit{Htol} &= 10^{-6}, \\ \mathit{ndense} &= 10, \\ \mathit{factol} &= 0.01. \end{aligned}$$

Note that small diagonals of H_N lead to large entries in S, while "dense" columns in N lead to excessive density in S. We therefore use *Htol* and *ndense* to control the partitioning of K in the following way. The j-th column of H is included in H_N (and column a_j of A is included in N) if

1. $H_{jj} \geq \mathit{Htol}\, \|a_j\|$, and
2. a_j has fewer than *ndense* nonzeros.

Since we scale A to give $\|a_j\| \approx 1$ for all j, we avoid storing an n-vector of norms by simplifying the first test to just $H_{jj} \geq \mathit{Htol}$. Including $\|a_j\|$ would give slightly greater reliability.

The third parameter *factol* is used as the stability tolerance u [DR82, DR83] when the *Factor* phase of MA27 is applied to K_B. In the extreme case (N void, $K_B = -AH^{-1}A^T$), *factol* is inoperative since MA27 is then performing Cholesky factorization on a (negative) definite matrix. In all other cases, *factol* affects the stability of the numerical factorization and the fill-in in the factors (beyond that predicted by the MA27 *Analyze*).

4.1. Iterative Refinement; Tightening Tolerances

Whenever a KKT system of the form

$$K\begin{pmatrix}\Delta x\\ -\Delta\pi\end{pmatrix} = \begin{pmatrix}w\\ r\end{pmatrix}$$

is solved (via a reduced KKT system), we estimate whether the tolerances *Htol* and *factol* are too lax by computing the residuals for the full system:

$$\begin{pmatrix}q_1\\ q_2\end{pmatrix} = \begin{pmatrix}w\\ r\end{pmatrix} - K\begin{pmatrix}\Delta x\\ -\Delta\pi\end{pmatrix} = \begin{pmatrix}w - H\Delta x + A^T\Delta\pi\\ r - A\Delta x - \delta\Delta\pi\end{pmatrix}. \tag{4.1}$$

Let the relative error in the residual be $\mathit{error} = (\|q_1\| + \|q_2\|)/(\|w\| + \|r\|)$.

If $\mathit{error} > 0.01$, we request that *Htol* be increased by a factor of 100 and *factol* be increased by a factor of 10, up to limits of 0.1 and 0.2 respectively. The partitioning of K and the factorization of the reduced KKT system are then repeated. If the tolerances are already at their limiting values, the program terminates with an error condition.

Otherwise, if $\mathit{error} > 10^{-4}$, we perform one step of iterative refinement:

$$K\begin{pmatrix}p_1\\ -p_2\end{pmatrix} = \begin{pmatrix}q_1\\ q_2\end{pmatrix}, \qquad \begin{pmatrix}\Delta x\\ \Delta\pi\end{pmatrix} \leftarrow \begin{pmatrix}\Delta x\\ \Delta\pi\end{pmatrix} + \begin{pmatrix}p_1\\ p_2\end{pmatrix}.$$

If the new relative error satisfies $\mathit{error} > 10^{-4}$, we request increased tolerances as in the previous paragraph.

If $\mathit{error} \le 10^{-4}$ either before or after refinement, the solve is taken to be sufficiently accurate.

The predictor-corrector algorithm uses more than one solve to obtain a search direction. Forming the products $A\Delta x$ and $A^T\Delta\pi$ in (4.1) is moderately expensive, but at least for the last solve, these vectors can be saved and used to update the residuals r and t during the linesearch:

$$\begin{aligned} r &\leftarrow r - \alpha_x(A\Delta x) - \alpha_z\delta^2\Delta\pi,\\ t &\leftarrow t + \alpha_x(Q + \gamma^2 I)\Delta x - \alpha_z(A^T\Delta\pi + \Delta y - \Delta z). \end{aligned}$$

Direct computation of $r = b - Ax - \delta^2\pi$ and $t = c + Qx + \gamma^2 x - A^T\pi - z + y$ can then be carried out less often—say every 10 iterations, or whenever $\|r\|$ or $\|t\|$ drops significantly.

Note that it is most effective to use iterative refinement on the full KKT system as described, not on the reduced system. In particular, *even implementations based on* $AH^{-1}A^T$ should compute corrections for both Δx and $\Delta\pi$.

5. Numerical Examples

To illustrate some numerical values arising in the reduced KKT systems of Section 3, we apply the basic primal-dual algorithm to two LP problems of the form

$$\min\ c^T x \text{subject to} Ax = b, \quad x \geq 0,$$

with

$$A = \begin{pmatrix} 1 & 1 & 3 & 3 \\ 1 & 2 & 1 & 2 \end{pmatrix}, \qquad c = \begin{pmatrix} 1 & 1 & 0 & 0 \end{pmatrix}^T,$$

using MATLAB$^{\text{TM}}$ [MLB87] with about 16 digits of precision.

5.1. A Non-degenerate LP

We first let the right-hand side and optimal solution be

$$b = \begin{pmatrix} 6 \\ 3 \end{pmatrix}, \qquad x^* = \begin{pmatrix} 0 & 0 & 1 & 1 \end{pmatrix}^T.$$

At the start of the fifth iteration of the primal-dual algorithm, we have $x = (3.9\text{e}{-6},\ 3.2\text{e}{-6},\ 1.000005,\ 0.999992)$, $z = (0.70,\ 0.70,\ 2.0\text{e}{-6},\ 1.7\text{e}{-6})$, and

$$K = \begin{pmatrix} 1.8\text{e+}5 & & & & 1 & 1 \\ & 2.2\text{e+}5 & & & 1 & 2 \\ & & 2.0\text{e}{-6} & & 3 & 1 \\ & & & 1.7\text{e}{-6} & 3 & 2 \\ 1 & 1 & 3 & 3 & & \\ 1 & 2 & 1 & 2 & & \end{pmatrix} = \begin{pmatrix} H & A^T \\ A & \end{pmatrix},$$

where $H = X^{-1}Z$. Recall that *Htol* defines which diagonals of H are considered large enough to form a block pivot. In terms of conventional error analysis for Gaussian elimination, *Htol* $= 1$ or 0.1 should be "safe", while *Htol* $< 10^{-3}$ (say) is likely to be unreliable. Various values of *Htol* give the following reduced KKT systems:

Htol	K_B	cond(K_B)
0.1	$\begin{pmatrix} 2.0\text{e}{-6} & & 3 & 1 \\ & 1.7\text{e}{-6} & 3 & 2 \\ 3 & 3 & -1\text{e}{-5} & -1\text{e}{-5} \\ 1 & 2 & -1\text{e}{-5} & -2\text{e}{-5} \end{pmatrix}$	7.5
1.8e−6	$\begin{pmatrix} 1.7\text{e}{-6} & 3 & 2 \\ 3 & -4\text{e+}6 & -1\text{e+}6 \\ 2 & -1\text{e+}6 & -5\text{e+}5 \end{pmatrix}$	5e+6
1e−20	$\begin{pmatrix} -1\text{e+}7 & -5\text{e+}6 \\ -5\text{e+}6 & -3\text{e+}6 \end{pmatrix}$	58

The large diagonals of H make K seem rather ill-conditioned ($\text{cond}(K) = 10^5$), but pivoting on those diagonals ($Htol = 0.1$) gives a very favorable reduced system: $\text{cond}(K_B) = 7.5$. Allowing one small pivot ($Htol = 1.8\text{e–}6$) gives the expected large numbers and high condition: $\text{cond}(K_B) \approx 10^6$. A second small pivot would normally have a similar effect, but here we have $m = 2$. The large numbers arising from m small pivots happen to form a very *well*-conditioned reduced system: $\text{cond}(K_B) = \text{cond}(AH^{-1}A^T) = 58$.

In general, the structure of K is such that pivoting on *any* nonzero diagonals of H should be safe if the following conditions hold:

- There are m *or more* small pivots of similar size (to within one or two orders of magnitude).
- The associated m or more columns of A form a well-conditioned matrix.

Unfortunately, in the presence of primal degeneracy there will be *less than* m small pivots, as the next example shows.

5.2. A Degenerate LP

Now let the right-hand side and optimal solution be

$$b = \begin{pmatrix} 3 \\ 2 \end{pmatrix}, \qquad x^* = \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}^T.$$

At the start of the sixth iteration of the primal-dual algorithm, we have
$x = (3.6\text{e–}7,\ 6.0\text{e–}7,\ 9.3\text{e–}7,\ 0.9999987)$, $z = (0.61,\ 0.32,\ 0.29,\ 2.2\text{e–}7)$, and

$$K = \begin{pmatrix} 1.7\text{e+}6 & & & & 1 & 1 \\ & 5.3\text{e+}5 & & & 1 & 2 \\ & & 3.1\text{e+}5 & & 3 & 1 \\ & & & 2.2\text{e–}7 & 3 & 2 \\ 1 & 1 & 3 & 3 & & \\ 1 & 2 & 1 & 2 & & \end{pmatrix}.$$

Two representative values of *Htol* give the following reduced KKT systems:

Htol	K_B	$\text{cond}(K_B)$
0.1	$\begin{pmatrix} 2.2\text{e–}7 & 3 & 2 \\ 3 & -3\text{e–}5 & -1\text{e–}5 \\ 2 & -1\text{e–}5 & -1\text{e–}5 \end{pmatrix}$	8e+5
1e–20	$\begin{pmatrix} -4\text{e+}7 & -3\text{e+}7 \\ -3\text{e+}7 & -2\text{e+}7 \end{pmatrix}$	1e+13

We see that the partially reduced system has a considerably lower condition than the fully reduced system. After one further iteration, the contrast is even greater (see the second table below).

5.3. Condition Numbers at Each Iteration

To further illustrate the effect of small H pivots, we list the condition of the reduced KKT systems arising at each iteration of the primal-dual algorithm with various values of *Htol*. The barrier parameter μ does not appear in K, but it is listed for reference.

For the first (non-degenerate) problem, the following condition numbers $\mathrm{cond}(K_B)$ were obtained:

Itn	μ	*Htol*:	0.1	1e–3	1e–4	1.8e–6	1e–20
1	3e–2		14	14	14	14	14
2	2e–3		72	300	300	300	300
3	2e–4		8	59	59	59	59
4	2e–6		7	7	5e+4	74	74
5	2e–8		7	7	7	5e+6	59
6	2e–10		7	7	7	7	7
7	2e–12		7	7	7	7	7

We see that allowing pivots as small as $H_{jj} = 10^{-p}$ is likely to give $\mathrm{cond}(K_B) = 10^p$ at some stage, except in the fortuitous case where are there are m or more small pivots.

For the degenerate problem, the following values of $\mathrm{cond}(K_B)$ were obtained:

Itn	μ	*Htol*:	0.1	1e–4	1e–20
1	2e–2		14	14	14
2	2e–4		23	23	23
3	6e–5		11	1e+3	1e+3
4	2e–5		114	114	7e+6
5	2e–7		8e+3	8e+3	1e+9
6	2e–9		8e+5	8e+5	1e+13
7	2e–11		8e+7	8e+7	2e+16
8	2e–13		8e+9	8e+9	∞

We see that very small pivots allow the condition of K_B to deteriorate seriously. We cannot expect a method based on $AH^{-1}A^T$ to make meaningful progress on this example beyond the sixth iteration. Since degeneracy is a feature of most real-life problems, it seems clear that small H pivots must be avoided if stability is to be assured.

To date, implementations based on $AH^{-1}A^T$ [LMS90] or some other "unstable" factorization [VC91] appear capable of attaining 8 digits of precision on most real-life applications, but occasionally attain only 6 digits or less. This is commendable performance, since 6 digits is undoubtedly adequate for most practitioners. It is the "occasionally less" that we maintain some concern about!

6. Implementation Details

The remaining discussion concerns our present implementation as it applies to LP problems ($Q = 0$). When γ or δ is nonzero in (2.1) the problem being solved is in fact a QP. A diagonal Q could easily be incorporated, as in [CLMS90]. The implementation is called PDQ1 (Primal-Dual QP code, version 1). We intend to allow a more general sparse Q in the future.

Various run-time parameters are used in PDQ1 to define starting points, stopping conditions, etc. (see below). Note that they are applied *after* the problem has been scaled. We assume that all computations are performed with about 16 digits of precision.

We do not perform any preprocessing of the data other than scaling. For example, we do not attempt to discard any rows or columns of A. (Nor do we attempt to fix variables on their bounds as the iterates converge.) In practice, of course, preprocessing can be very successful in reducing the problem dimensions and improving the numerical performance of solution algorithms. Our aim is to deal directly with the problem data and achieve reliability in the presence of redundant constraints, null variables, etc. (since preprocessors are not guaranteed to eliminate such difficulties).

We make an exception with regard to scaling, since we wish to solve an entire set of test problems with a single set of run-time parameters. Without scaling, excessively cautious parameter values may be needed to achieve reasonable performance.

6.1. Scaling

Row and column scales are first determined by an iterative procedure that tends to make the elements of A close to one [Fou82]. An “effective right-hand side” v is then defined according to

$$v = b - \sum_{l_j = u_j} a_j l_j - \sum_{l_j > 0} a_j l_j - \sum_{u_j < 0} a_j u_j, \tag{6.1}$$

and the row scales are applied to v to obtain the quantities $\bar{v}$ and $\sigma \equiv \|\bar{v}\|$. If $\sigma > 1$, all row and column scales are multiplied by σ.

This is the scaling procedure used in MINOS [MS87]. In most cases it has the effect of making $\|\bar{x}^*\| \approx 1$, where $\bar{x}^*$ is the solution of the scaled problem. As noted elsewhere [GMSW89, Marx89], the test problems *grow7*, *grow15* and *grow22* are exceptions in that $\|\bar{x}^*\| \approx 10^7$. To assist such cases we have implemented an additional scaling that takes effect if $v = 0$ in (6.1).

The *grow* problems happen to be of the form

$$\min \quad c^T x \text{subject to} Ax = 0, \quad 0 \le x \le u.$$

We first note that the optimal solution satisfies $c^T x^* \le 0$ (since $x = 0$ is a feasible point). Assuming $x^* \ne 0$, we can then say that $x_j^* = u_j$ for at least one j (since if some feasible point satisfies $0 \le x < u$, the point ρx is also feasible for some $\rho > 1$ and it has an improved objective value).

It follows that if $\tau \equiv \min_j u_j > 1$, a smaller $\|\bar{x}^*\|$ will result if all the scales are multiplied by τ.

For the *grow* problems we found that $\tau \approx 3000$, and the additional scaling by τ reduced $\|\bar{x}^*\|$ from 10^7 to 10^4 and improved the reliability of the solves with K. The effective right-hand side v was zero for one other problem (*sc205*). In this case, $\tau = 100$ and the extra scaling reduced the iteration count from 15 to 11.

6.2. Scaling c

Once scale factors are obtained as above, they are applied to the data A, b, c, l, u. A further scale factor is then applied to c to make $\|c\| \approx 1$.[3] In most cases the effect is to make $\|\bar{\pi}^*\| \approx 1$ for the scaled problem.

6.3. Starting

The initial values for the scaled primal and dual variables were chosen as follows (with $\xi_0 = \zeta_0 = 1$). Recall that the initial values of x and π do not affect the subsequent iterations (except that x is used to define the initial s_1 and s_2).

- $x_j = 0$ if zero lies between the bounds; otherwise, $x_j = l_j$ or u_j, whichever bound is nearest zero.
- $(s_1)_j = \max(\xi_0, x_j - l_j)$; $\quad (s_2)_j = \max(\xi_0, u_j - x_j)$.
- $\pi = 0$.
- $y_j = z_j = \zeta_0$.

The initial value of μ was set to balance the parts of the residual vector in (2.2) that do and do not depend on μ (with $\mu_0 = 0.1$):

- $\mu = \mu_0 \|f_0\|_2 / \sqrt{nbound}$,

where *nbound* ($\le 2n$) is the number of finite upper and lower bounds.

6.4. Stopping

If (x, s_1, s_2) and (π, z, y) are primal and dual feasible respectively, the duality gap (the difference between the primal and dual objectives) is

$$s_1^T z + s_2^T y = (c^T x + \tfrac{1}{2} x^T \widetilde{Q} x + \tfrac{1}{2} p^T p) - (b^T \pi + l^T z - u^T y - \tfrac{1}{2} x^T \widetilde{Q} x - \tfrac{1}{2} p^T p),$$

where $\widetilde{Q} = Q + \gamma^2 I$. The stopping criterion for LP problems required the following, with $\delta_{\text{fea}} = \delta_{\text{opt}} = 10^{-d}$ meaning a request for d digits of accuracy ($d = 6$ or 8):

- $\|r\|_2 / (1 + \|x\|) \le \delta_{\text{fea}}$.
- $\|t\|_2 / (1 + \|\pi\|) \le \delta_{\text{fea}}$.
- $(s_1^T z + s_2^T y) / (1 + |c^T x|) \le \delta_{\text{opt}}$.

After each iteration, a minimum value of μ is defined in terms of the objective function and the optimality tolerance. In the LP case,

- $\mu_{\min} = (1 + |c^T x|) \delta_{\text{opt}} / (10\,nbound)$.

If the current μ is below $2\mu_{\min}$, then μ is not reduced for the next iteration.

[3]The Euclidean norm $\|\,.\,\|_2$ is required for terms in the merit function $\|f_\mu\|^2$. For other vectors v of length n we define $\|v\| \equiv \sum |v_j| / \sqrt{n}$ (which approximates $\|v\|_2$ but is cheaper to evaluate). Since c may be sparse, we scale it by $\sum |c_j| / \sqrt{n_c}$, where n_c is the number of nonzeros in c. Using n_c in place of n affected the iteration counts for many of the test problems—on average favorably.

6.5. Regularization

The values $\gamma = 10^{-5}$ and $\delta = 10^{-5}$ were used, with seemingly satisfactory results. Larger values may perturb the solution too much, and smaller values can lead to near-singularity in K and perhaps divergence of the iterates.

Recall that the regularization terms in the objective are $\frac{1}{2}\|\gamma x\|^2 + \frac{1}{2}\|\delta\pi\|^2$, with $\|x\| \approx 1$ and $\|\pi\| \approx 1$ near a solution. For many problems we have observed that $\|\pi\|$ decreases sharply in the final primal-dual iterations, showing that a nonzero δ helps resolve some ambiguity in the dual solution.

In line with the theory of [MM79, Man84], there is no similar decrease in $\|x\|$ when γ is rather small.

6.6. Solving the KKT Systems

To date we have used the Harwell Subroutine Library package MA27 [DR82, DR83] to solve the reduced KKT systems. This is a multifrontal code designed to perform well on vector machines on matrices that are definite or nearly definite (i.e., most eigenvalues have the same sign).

When there are no free variables or dense columns, the tolerances are such that $K_B = -AH^{-1}A^T$ for most of the early iterations. The performance should then be similar to other Cholesky-based implementations.

As the optimal solution is approached, many diagonals of H become small and K_B becomes more indefinite as its dimension increases. In some cases the MA27 *Factor* generates substantially more nonzeros than predicted by the *Analyze*, and the iteration time deteriorates significantly.

As always, improvements in computation time will come from speeding up the KKT solves. A major modification of MA27 is being developed as MA47 [DGRST89], and we expect that its performance in the KKT context will be considerably improved. A promising alternative is the code described recently in [FM91].

7. Numerical Results

In this section we present results obtained from the *netlib* collection of LP test problems [Gay85]. One aim is to explore the probable dimensions of the reduced KKT systems that must be solved (and determine how often a new *Analyze* is needed).

Another aim is to show that a primal-dual barrier algorithm similar to the ones in [Meh89, Meh90, LMS89, LMS90] can achieve comparably low iteration counts without the benefit of preprocessing (other than scaling) and with relatively simple starting conditions and a straightforward reduction of μ each iteration. This is partly due to the improved reliability of the numerical linear algebra in the presence of free variables, dense columns, and near-singularity.

During the code development, occasional high iteration counts were usually found to be the result of lax tolerances in forming and solving the reduced KKT systems (just as a simplex code could be expected to iterate indefinitely if an unreliable basis package were used). With the current MA27 factorizer, it remains desirable to use lax tolerances tentatively (to enhance sparsity), since they are often adequate. Provision for iterative refinement and tightening of the factorization tolerances (Section 4.1) seems to provide a reliable safeguard.

As an example, most of the test problems solved successfully with the tolerances fixed at $Htol = 10^{-8}$ and $factol = 0.001$. This is *extremely* lax in terms of conventional Gaussian elimination, but note that implementations based on $AH^{-1}A^T$ are effectively using $Htol = 0$ and $factol = 0$, with no increase possible. Iterative refinement can again be invoked, but that alone may be unsuccessful.

7.1. Dimension of the Reduced KKT Systems

Let n_k be the dimension of the reduced KKT system K_B (3.3) at iteration k, and let $r_k = n_k/m$. The first graph in Figure 1 plots the ratios r_k for a representative selection of problems (using $Htol = 10^{-6}$ and requesting 8 digits of accuracy). The name of each problem appears near the end of the associated plot.

The value $r_k = 1$ implies that $K_B = AH^{-1}A^T$ at iteration k. For example, problems *scsd6* and *ship12l* both give fully reduced systems at the beginning, since all elements of H are of order 1 initially, and there are no dense columns. In contrast, $r_k \approx 2$ for most of the *pilots* iterations, because almost m columns of A contain 10 or more nonzeros and are included in B throughout.

In general, r_k stays almost constant until the final iterations, when many diagonals of H start falling below $Htol$. The dimension of K_B increases as more columns of A are included in B.

Similarly, let nz_k be the number of nonzeros in the MA27 factorization of K_B at iteration k, and let $\bar{r}_k = nz_k/nz_1$. (The minimum number of nonzeros happens to occur at the first iteration.) The second graph in Figure 1 plots the ratios $\bar{r}_k$ for the same problems. The values $\bar{r}_k > 2$ represent a serious loss of sparsity in order to preserve stability.

Some observations follow.

- The dimension of K_B changes from its previous value rather more than half of the time. This determines how many times a new *Analyze* is needed. Some statistics are given in Table 1.

- The cpu time for an MA27 *Analyze* is usually moderate compared to a *Factor*, but sometimes it can be substantial. Table 1 shows how much time is spent in each phase, as a percentage of the total cpu time. (There is normally one *Factor* per iteration, except on rare occasions when the stability tolerances are tightened.)

- There is typically a sharp increase in the MA27 *Factor* nonzeros during the final iterations. In particular, requesting 8 digits of accuracy rather than 6 carries a substantial cost.

These matters reflect the cost of stability compared to implementations based on $AH^{-1}A^T$ (for which $r_k = \bar{r}_k = 1$ throughout).

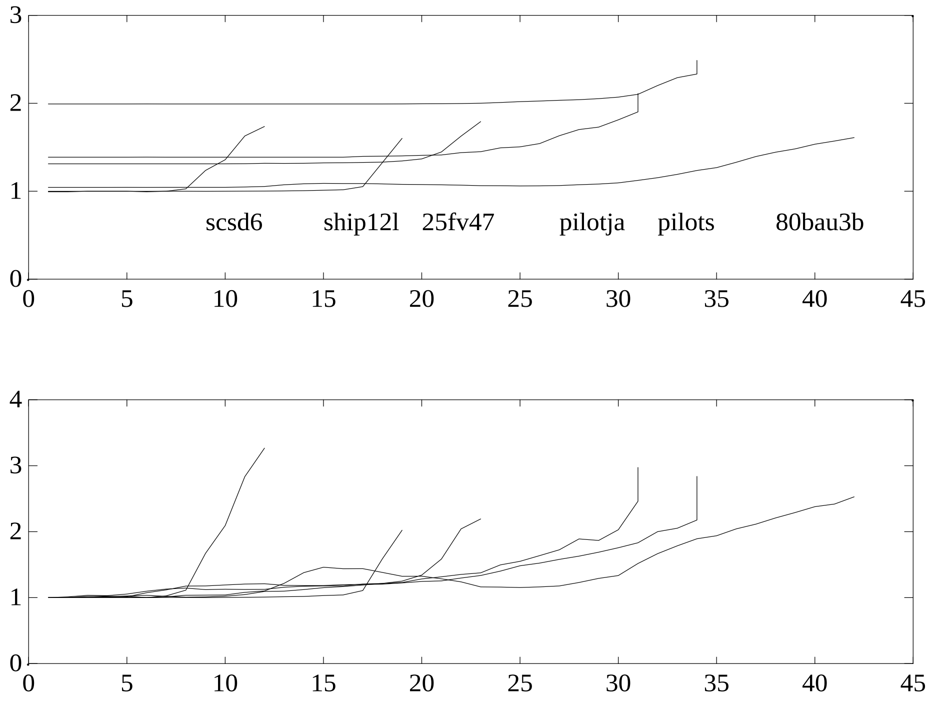

Figure 1: The dimension of the reduced KKT systems K_B (relative to m), and the number of nonzeros in the MA27 factors (relative to the first factorization).

7.2. Performance on the *netlib* Test Set

Table 2 lists the iteration counts for PDQ1 in solving the first 70 LP test problems in *netlib*. We give results for both 6 and 8-digit accuracy. They are compared to the iteration counts recorded by OB1 in [LMS90], where 8-digit accuracy was also requested (and in most cases obtained). OB1 used some preprocessing of the data [LMS89]. The column labeled "Diff" indicates significant differences between columns 1 and 3. A + means PDQ1 required more iterations than OB1.

Some observations follow.

- The stability tolerances $Htol = 10^{-6}$ and $factol = 0.01$ proved to be reliable in almost all cases. The *grow* problems, *scsd8*, *pilotja* and *pilots* required iterative refinement at certain points and ultimately made one request for stricter tolerances ($Htol = 10^{-4}$ and $factol = 0.1$).

- The *grow* problems illustrate the effect of inaccurate solution of the KKT system. For the last few iterations there were symptoms of difficulty (refinement, reversion to pure primal-dual, backtracking in the linesearch) and stricter tolerances were finally requested for the last iteration. When $Htol = 10^{-6}$ and

	Analyze	*Factor*	*Analyze* time	*Factor* time
scsd6	11	12	20%	32%
ship12l	14	19	18%	30%
25fv47	14	23	13%	72%
pilotja	19	32	16%	75%
pilots	20	35	15%	81%
80bau3b	38	42	29%	45%
degen3	9	28	29%	66%

Table 1: The number of MA27 *Analyze* and *Factor* calls, and the percentage of time spent in each.

factol = 0.1 were used from the beginning, no symptoms of numerical error arose and the iteration counts were 19, 21 and 21 (a significant improvement for the last two cases, though still more than achieved by OB1).

- It is not clear why significantly more iterations were required for *cycle*, *wood1p* and *woodw*. These problems are from the same source and probably have a distinguishing characteristic that would explain the OB1 advantage.
- The first three *pilot* problems solved in significantly fewer iterations than with OB1. About 80 free variables and some notoriously narrow bounds ($l_j = 0$, $u_j = 10^{-5}$) do not appear to have caused difficulty.
- Otherwise, we see that the iteration counts for OB1 and PDQ1 are comparable for most problems.
- In terms of cpu time, the results would be far from comparable. We defer such statistics until a more efficient indefinite solver is installed.

8. Conclusions

For various good reasons, most interior-point codes for LP (and for separable QP) have been based on Cholesky factors of matrices of the form $AH^{-1}A^T$. Excellent performance has been achieved (notably by [LMS90, Meh90, CLMS90]) and the primary sources of difficulty have been thought to be dense columns and free variables.

For general QP problems a full KKT system must be solved [Pon90], and it is becoming increasingly recognized that such an approach removes the above-mentioned difficulties (e.g. [Tur90, GMPS90, FM91, Van91]).

Here we have emphasized the fact that the real source of numerical error lies in pivoting on small diagonals of H in the presence of primal degeneracy. Reducing a KKT system K to $AH^{-1}A^T$ is equivalent to pivoting on *all* diagonals of H, regardless of size. We suggest forming "reduced KKT systems" by pivoting on just the diagonals of H that are suitably large. This allows us to avoid factorizing a full KKT system, and often leads to use of $AH^{-1}A^T$ in the early iterations when it is numerically safe.

Code:	OB1	PDQ1	PDQ1	Diff
Digits:	8	6	8	
25fv47	25	23	23	
80bau3b	38	37	42	
adlittle	12	12	13	
afiro	9	9	10	
agg	24	19	21	
agg2	18	19	21	
agg3	17	19	21	
bandm	17	16	18	
beaconfd	10	18	21	++
bore3d	18	21	23	+
brandy	19	17	19	
capri	18	19	20	
cycle	30	37	43	+++
czprob	35	33	36	
degen2	14	15	17	
degen3	20	26	28	+
e226	22	18	20	
etamacro	29	19	23	–
fffff800	28	23	27	
forplan	21	23	25	
ganges	16	16	18	
gfrdpnc	18	17	18	
greenbea	41	48	51	++
greenbeb	33	34	38	+
grow7	14	17	19	+
grow15	16	18	26	++
grow22	16	18	30	++
israel	23	22	23	
kb2	15	13	13	
nesm	30	23	30	
pilotja	46	28	31	–––
pilotwe	46	30	33	–––
pilot4	36	25	27	––
pilotnov	20	16	17	
pilots	29	31	34	+
recipe	10	13	15	+
sc205	11	11	13	
scagr25	16	16	17	
scagr7	12	13	14	
scfxm1	17	17	18	
scfxm2	19	19	20	
scfxm3	20	19	21	
scorpion	14	12	14	
scrs8	27	20	22	–
scsd1	11	9	9	
scsd6	12	11	12	
scsd8	10	11	15	+
sctap1	15	14	15	
sctap2	20	15	15	–
sctap3	17	16	16	
seba	19	18	20	
share1b	20	17	18	
share2b	12	11	12	
shell	21	20	21	
ship04l	15	17	18	
ship04s	15	17	18	
ship08l	16	16	18	
ship08s	14	16	17	
ship12l	18	17	19	
ship12s	18	17	18	
sierra	18	18	20	
stair	16	17	17	
standata	15	17	19	
standmps	24	24	25	
stocfor1	19	13	13	–
stocfor2	22	33	33	++
tuff	19	24	27	+
vtpbase	13	15	18	+
wood1p	14	27	29	+++
woodw	20	26	30	++

Table 2: Iteration counts for OB1 (requesting 8 digits) and PDQ1 (6 and 8 digits).

The primary advantage is intended to be numerical reliability. The drawbacks are that a new *Analyze* is required each time the reduced KKT system changes in dimension, and that we are dependent on the efficiency of a symmetric indefinite factorizer. Some new codes [DGRST89, FM91] promise to narrow the gap between indefinite and definite solvers. A variant of "reduced KKT systems" has recently been given in [Van91]. An advantage is that it avoids the need for an indefinite solver, but in its present form it is susceptible to the normal dangers of small pivots in H.

In terms of overall strategy, it remains to be seen which of the approaches in [LMS90, Tur90, FM91, Van91] and the present paper will offer the most favorable balance between efficiency and reliability.

Acknowledgement

We are grateful to Dr. Florian Jarre for many helpful discussions on this work during his postdoctoral year at the Department of Operations Research.

References

[AHRT90] K. M. Anstreicher, D. den Hertog, C. Roos and T. Terlaky (1990). A long-step barrier method for convex quadratic programming, Report 90-53, Faculty of Technical Mathematics and Informatics, Delft University of Technology, Delft, The Netherlands.

[BK77] J. R. Bunch and L. Kaufman (1977). Some stable methods for calculating inertia and solving symmetric linear systems, *Mathematics of Computation* 31, 162–179.

[BP71] J. R. Bunch and B. N. Parlett (1971). Direct methods for solving symmetric indefinite systems of linear equations, *SIAM Journal on Numerical Analysis* 8, 639–655.

[CLMS90] T. J. Carpenter, I. J. Lustig, J. M. Mulvey and D. F. Shanno (1990). A primal-dual interior-point method for convex separable nonlinear programs, Report SOR 90-2, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.

[DG85] J. J. Dongarra and E. Grosse (1985). Distribution of mathematical software via electronic mail, *SIGNUM Newsletter* 20, 45–47.

[DGRST89] I. S. Duff, N. I. M. Gould, J. K. Reid, J. A. Scott and K. Turner (1989). The factorization of sparse symmetric indefinite matrices, CSS Report 236, Computer Science and Systems Division, AERE Harwell, Oxford OX11 0RA, England.

[DR82] I. S. Duff and J. K. Reid (1982). MA27: A set of Fortran subroutines for solving sparse symmetric sets of linear equations, Report R-10533, Computer Science and Systems Division, AERE Harwell, Oxford, England.

[DR83] I. S. Duff and J. K. Reid (1983). The multifrontal solution of indefinite sparse symmetric linear equations, *ACM Transactions on Mathematical Software* 9, 302–325.

[FM68] A. V. Fiacco and G. P. McCormick (1968). *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, John Wiley and Sons, New York and Toronto.

[Fou82] R. Fourer (1982). Solving staircase linear programs by the simplex method, 1: Inversion, *Mathematical Programming* 23, 274–313.

[FM91] R. Fourer and S. Mehrotra (1991). Performance of an augmented system approach for solving least-squares problems in an interior-point method for linear programming, Preliminary Report, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL.

[Gay85] D. M. Gay (1985). Electronic mail distribution of linear programming test problems, *Mathematical Programming Society COAL Newsletter*, December 1985.

[GMPS90] P. E. Gill, W. Murray, D. B. Ponceleón and M. A. Saunders (1990). Preconditioners for indefinite systems arising in optimization, Report SOL 90-8, Department of Operations Research, Stanford University, Stanford, CA.

[GMPS91] P. E. Gill, W. Murray, D. B. Ponceleón and M. A. Saunders (1991). Primal-dual methods for linear programming, Report SOL 91-3, Department of Operations Research, Stanford University, Stanford, CA.

[GMSTW86] P. E. Gill, W. Murray, M. A. Saunders, J. A. Tomlin and M. H. Wright (1986). On projected Newton barrier methods for linear programming and an equivalence to Karmarkar's projective method, *Mathematical Programming* 36, 183–209.

[GMSW89] P. E. Gill, W. Murray, M. A. Saunders and M. H. Wright (1989). A practical anti-cycling procedure for linearly constrained optimization, *Mathematical Programming* 45, 437–474.

[Grc90] J. F. Grcar (1990). Matrix stretching for linear equations, Report SAND90-8723, Sandia National Laboratories, Albuquerque, NM.

[JSS90] F. Jarre, G. Sonnevend and J. Stoer (1990). On the complexity of a numerical algorithm for solving generalized convex quadratic programs by following a central path, *Contemporary Mathematics* 114, 233–242.

[KLSW89] N. K. Karmarkar, J. C. Lagarias, L. Slutsman and P. Wang (1989). Power-series variants of Karmarkar-type algorithms, *AT&T Technical Journal* 68, 3, 20–36.

[KMY88] M. Kojima, S. Mizuno and A. Yoshise (1988). A primal-dual interior-point algorithm for linear programming, in N. Megiddo (ed.), *Progress in Mathematical Programming*, Springer-Verlag, NY, 29–48.

[LMS89] I. J. Lustig, R. E. Marsten and D. F. Shanno (1989). Computational experience with a primal-dual interior-point method for linear programming, Report SOR 89-17, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.

[LMS90] I. J. Lustig, R. E. Marsten and D. F. Shanno (1990). On implementing Mehrotra's predictor-corrector interior-point method for linear programming, Report SOR 90-3, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.

[LMC89] I. J. Lustig, J. M. Mulvey and T. J. Carpenter (1989). Formulating stochastic programs for interior-point methods, Report SOR 89-16, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.

[Man84] O. L. Mangasarian (1984). Normal solutions of linear programs, *Mathematical Programming Study* 22, 206–216.

[MM79] O. L. Mangasarian and R. R. Meyer (1979). Nonlinear perturbation of linear programs, *SIAM Journal of Control and Optimization* 17, 745–757.

[Marx89] A. Marxen (1989). *Primal Barrier Methods for Linear Programming*, Ph.D. Thesis, Department of Operations Research, Stanford University, Stanford, CA.

[MMS89] K. A. McShane, C. L. Monma and D. F. Shanno (1989). An implementation of a primal-dual interior point method for linear programming, *ORSA Journal on Computing* 1, 70–83.

[Meg86] N. Megiddo (1986). Pathways to the optimal set in linear programming, in N. Megiddo (ed.), *Progress in Mathematical Programming*, Springer-Verlag, New York, 131–158.

[Meh89] S. Mehrotra (1989). On finding a vertex solution using interior-point methods, Report 89-22, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL.

[Meh90] S. Mehrotra (1990). On the implementation of a primal-dual interior-point method, Report 90-03, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL.

[Meh91] S. Mehrotra (1991). Handling free variables in interior methods, Report 91-06, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL.

[MLB87] C. Moler, J. Little and S. Bangert (1987). PRO-MATLAB User's Guide, The MathWorks, Inc., Sherborn, MA.

[MA89] R. D. C. Monteiro and I. Adler (1989). Interior path-following primal-dual algorithms. Part II: Convex quadratic programming, *Mathematical Programming* 44, 43–66.

[MAR88] R. D. C. Monteiro, I. Adler and M. G. C. Resende (1988). A polynomial-time primal-dual affine scaling algorithm for linear and convex quadratic programming and its power-series extension, Report ESRC 88-8, Department of Industrial Engineering and Operations Research, University of California at Berkeley, Berkeley, CA.

[MS87] B. A. Murtagh and M. A. Saunders (1987). MINOS 5.1 User's Guide, Report SOL 83-20R, Department of Operations Research, Stanford University, Stanford, CA.

[Pon90] D. B. Ponceleón (1990). *Barrier Methods for Large-Scale Quadratic Programming*, Ph.D. Thesis, Computer Science Department, Stanford University, Stanford, CA.

[Roc76] R. T. Rockafellar (1976). Monotone operators and the proximal-point algorithm, *SIAM Journal on Control and Optimization* 14, 877–898.

[Set89] R. Setiono (1989). An interior dual proximal-point algorithm for linear programs, Report 879, Computer Sciences Department, University of Wisconsin, Madison, WI.

[Set90a] R. Setiono (1990). Interior proximal-point algorithm for linear programs, Report 949, Computer Sciences Department, University of Wisconsin, Madison, WI.

[Set90b] R. Setiono (1990). Interior dual least 2-norm algorithm for linear programs, Report 950, Computer Sciences Department, University of Wisconsin, Madison, WI.

[Set90c] R. Setiono (1990). Interior dual proximal point algorithm using preconditioned conjugate gradient, Report 951, Computer Sciences Department, University of Wisconsin, Madison, WI.

[Tur90] K. Turner (1990). Computing projections for the Karmarkar algorithm, Report 49, Department of Mathematics and Statistics, Utah State University, UT.

[Van90a] R. J. Vanderbei (1990). ALPO: Another Linear Program Solver, Technical Report, AT&T Bell Laboratories, Murray Hill, NJ.

[Van90b] R. J. Vanderbei (1990). Splitting dense columns in sparse linear systems, Technical Report, AT&T Bell Laboratories, Murray Hill, NJ.

[Van91] R. J. Vanderbei (1991). Symmetric quasi-definite matrices, Report SOR 91-10, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.

[VC91] R. J. Vanderbei and T. J. Carpenter (1991). Symmetric indefinite systems for interior-point methods, Report SOR 91-7, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.